

# **A Trainable Text Summarizer Employing Statistical and Linguistic Features**

Joel Larocca Neto      Alex A. Freitas      Celso A. A. Kaestner

Pontifical Catholic University of Paraná  
Rua Imaculada Conceição, 1155  
80.215-901 Curitiba – PR - BRAZIL  
{joel, alex, kaestner}@ppgia.pucpr.br

**Paper ID:** P

**Keywords:** Text Summarization, Trainable Summarizers, Text-mining, Information Retrieval, Machine Learning.

**Contact Author:** Celso A. A. Kaestner

**Under consideration for other conferences (specify)?** No.

## **Abstract**

In this work we address the automatic summarization task. According to the literature, extractive-summary generation depends basically on heuristics; however, there are few indicatives of how to select the relevant features. We will present a trainable summarization procedure which employs linguistic and statistical features, extracted directly and automatically from the original text. Computational results obtained with the application of the proposed summarizer to some well known text databases are presented, and we compare these results to some baseline summarization procedures.

# A Trainable Text Summarizer Employing Statistical and Linguistic Features

Paper-ID: P

## Abstract

In this work we address the automatic summarization task. According to the literature, extractive-summary generation depends basically on heuristics; however, there are few indicatives of how to select the relevant features. We will present a trainable summarization procedure which employs linguistic and statistical features, extracted directly and automatically from the original text. Computational results obtained with the application of the proposed summarizer to some well known text databases are presented, and we compare these results to some baseline summarization procedures.

## 1 Introduction

*Text-Mining (TM)* is a research field related to *Data-Mining (DM)* (Weiss 98) which treats basically free text with little or no semantic information about the data. An important *TM* task is summarization, which consists of reducing the size of a text while preserving its information content (Luhn 58), (Spark-Jones 99).

Summary construction is a complex task and ideally requires deep natural language processing capacities which are beyond the ability of currently-known techniques (Mitra 97). In order to simplify the problem, current research focus on extractive summaries, defined as a subset of the sentences of the original text. These summaries do not guarantee a good narrative coherence, but are useful for relevance judgement.

Typically, the summary is employed to answer specific questions about the original text, or is used as a pointer to some part of the original document. Obviously a very important advantage of using an extractive summary is its

reduced reading time. Automatic summary generation has some advantages: (i) the size of the summary can be controlled; (ii) its contents is determinist; and (iii) the relation between a sentence in the summary and the original text block in is easily established.

In our work we address the automatic summarization task. Published research work on extractive-summary generation depends basically on heuristics (Edmundson 69), (Kupiec 95); however, few indicatives are given of how to select the relevant features. We will present a summarization procedure which employs some linguistic and statistical features, extracted directly and automatically from the original text.

The rest of the paper is organized as follows: section 2 describes the text summarization task; section 3 presents the features employed in the proposed algorithms; in section 4 we describe the proposed trainable summarizer; section 5 presents the obtained computational results; and finally, section 6 presents some conclusions and future research work.

## 2 Text summarization: a review

According to Spark-Jones (Spark-Jones 99) a summarization process can be separated in three steps: (1) the preprocessing step, where a structured representation of the original text is generated; (2) the transformation of this structure into a summary representation; and (3) the generation of the summary from the summary representation.

The preprocessing step aims to reduce the dimensionality of the representation space, and usually includes: (i) *stop-word* elimination – common words which carry no semantics and do not aggregate relevant information to the *TM* task are eliminated; (ii) *case folding*: consists of converting all the characters to the same kind of letter case - either upper case or lower case; (iii) *stemming*: syntactically-similar words, such as plurals, verbal variations, etc. are considered similar; the purpose of this procedure is to

obtain the stem or radix of each word, which characterize its semantic.

A frequently employed text model is the vectorial model (Salton 97). After the preprocessing step each text element – typically a sentence in the case of text summarization – corresponds to an  $N$ -dimensional vector. In this metric space an adequate metric can be easily established: the most employed one is the cosine measure, defined as  $\cos \bullet = (\langle x, y \rangle) / (|x| \cdot |y|)$  for vectors  $x$  and  $y$ , and where  $\langle \cdot, \cdot \rangle$  indicates the scalar product, and  $|x|$  stands for the module of  $x$ . Therefore maximum similarity is indicated by  $\cos \bullet = 1$ , whereas  $\cos \bullet = 0$  indicates total discrepancy between the text elements.

A key point in summarization research is how to evaluate the quality of the generated summary. A detailed evaluation of summarizers was made at the TIPSTER Text Summarization Evaluation Conference (SUMMAC) (Mani 98a), as part of an effort to standardize summarization test procedures. In these tests summaries of the reference database collection were provided by human judges, allowing a direct comparison of the performance of the systems that participated in the conference. However, the human effort to elaborate such summaries is huge. In the case of summaries generated from different human judges, there is low concordance: only 46 % according to Mitra (Mitra 97); more importantly: summaries produced by the same human judge in different dates have only 55 % of agreement (Rath 61).

If we consider the existence of a “reference summary” the classical *IR precision* and *recall* measures can be employed to evaluate the quality of the automatically-generated summary: a sentence will be called *correct* if it belongs to the reference summary. As usual, *precision* is the number of selected correct sentences divided by the total number of sentences selected, and *recall* is the ratio of the number of selected correct sentences over the total number of correct sentences. In the case of fixed length summaries the two measures will be identical.

An intelligent solution which allows to obtain the reference extractive summary was proposed by Mani (Mani 98b): if each original text contains an author-provided summary, the corresponding size- $K$  reference extractive summary for this document consists of the  $K$  most similar sentences to the author-provided summary, according to the vectorial model. This approach is adequate for automatic procedure,

and strongly facilitate the generation of reference extractive summaries.

Given a collection of documents and their corresponding reference extractive summaries, we can use a different class of algorithms: trainable summarizers. A trainable summarizer is a classical machine learning algorithm which employs, as training set, examples corresponding to the sentences of the documents to be summarized and a set of features describing those sentences. The trainable summarizer performs a classification task, as follows: each sentence of the training set is labeled as “correct” if it belongs to the extractive reference summary, or as “incorrect” otherwise; then the trainable summarizer is expected to “learn” the pattern which leads to the summaries, by identifying relevant feature values which are most correlated with the classes “correct” or “incorrect”; when a new document is given to the system, it uses the learned patterns to classify each sentence of that document into either a “correct” or “incorrect” sentence, therefore producing a new extractive summary. Obviously a crucial issue in this framework is how to obtain the relevant set of features.

### 3 Feature identification for trainable summarizers

In the literature of text-summarization we can find a large number of features that can be used to train a summarizer. These features can be roughly divided into six groups, as follows:

**(a) Sentence length.** This feature can be used, for instance, to penalize sentences that are too short; very short sentences are not expected to be selected for the summary. An example of this use is found in (Kupiec 95), where it is defined as a binary feature, whose boolean value depends on whether or not the number of words in a sentence is smaller than a given threshold.

**(b) Sentence position.** This feature can involve the position of a sentence in the document as a whole, the position of a sentence in a paragraph, etc. This kind of feature presented good results in several projects (Edmundson 69), (Kupiec 95), (Mani 98b), (Teufel 99). It should be noted that in general these projects assumed that the text was already segmented into paragraphs, so that the position of a sentence in a paragraph, or the presence of sections in the document (such

as “Introduction”, “Conclusions”, etc.) can be taken into account.

**(c) Thematic features.** We focus here on thematic features based on information retrieval techniques.

Since the seminal work of Luhn (Luhn 58), research on summarization has often used measures based on information retrieval (Edmundson 69), (Kupiec 95), (Teufel 99). Examples are the well-known measures of TF (term frequency) and TF-IDF (term frequency  $\times$  inverse document frequency) (Mani 98b).  $TF(w,d)$ , the term frequency of a word  $w$  in a document  $d$ , is the number of occurrences of  $w$  in  $d$ ;  $IDF(w)$ , the inverse document frequency of a word  $w$ , is normally defined as  $\log(|D|/DF(w))$ , where  $|D|$  is the number of documents in the document base and  $DF(w)$  is the number of documents in which  $w$  occurs; the  $TF-IDF(w,d)$  measure is the product of the  $TF(w,d)$  and  $IDF(w)$  measures. The higher this product, the more relevant word  $w$  is to document  $d$ , and the higher the weight that  $w$  will have in a formula used to measure the similarity between a query and a document.

In text summarization our “unit of work” is a single document, and we have to select a set of relevant sentences to be included in the summary out of all sentences in a given document. Hence, the notion of document in information retrieval can be replaced by the notion of sentence in text summarization and the “new” measure will be  $TF-ISF(w,s)$  ( $s$  used for sentence) (Larocca 00).

In this work thematic words are obtained by using trainable techniques, as discussed in (Turney 00) and (Nevill-Manning 99).

**(d) Text cohesion.** The basic idea of this feature is that sentences having a larger degree of cohesion are more relevant to be selected for the summary. A sentence’s degree of cohesion can be measured in several ways, such as computing the similarity of the sentence with every other sentence of the document, computing the similarity of the sentence with the document centroid, etc. (Barzilay 97), (Carbonell 98), (Mani 98b), (Mittra 97).

**(e) Rethorical structure of a text.** A complete analysis of the rethorical structure of a text would be too complex to be obtained, and it would require a full understanding of the text. However, some methods based on a superficial understanding of the text have been used to obtain good-quality summaries, as in (Teufel 99) and (Yaari 97).

**(f) Text Morphology.** Part-of-speech algorithms, such as the one proposed by Brill (Brill 92), determine the part-of-speech (noun, adjective, verb, etc.) of each word in a sentence. Intuitively, this analysis can be useful for text summarization, since there seems to be some correlation between the part-of-speech associated with some words and its relevance for summarization purposes; some evidence for this is presented by Carbonell (Carbonell 98).

## 4 Our proposal: a trainable summarizer employing statistical and linguistic features

Taking into account the kinds of features identified in the previous section, we propose a trainable summarizer. We use “statistics-oriented” features and “linguistics-oriented” features - loosely speaking.

### 4.1 Employed features

We use the following “statistics-oriented” features:

**(a) Sentence Length.** The normalized length of the sentence, which is the ratio of the number of words occurring in the sentence over the number of words occurring in the longest sentence of the document.

**(b) Sentence Position.** In our work we use the a generic positional feature proposed by Nevill-Manning (Nevill-Manning 99), whose value is defined as the percentile of the sentence in the entire document, which is normalized to take on values between 0 and 1.

**(c) Mean-TF-ISF.** The mean value of the  $TF-ISF$  measure for the words of the sentence. Recall that each sentence is represented by a vector of  $TF-ISF$  values, one value for each word.

**(d) Similarity to Title.** This feature is obtained by using the title of the document as a “query” against all the sentences of the document. First both the document title and each sentence of the document are converted to their vectorial representation; then the similarity between the title and each sentence is computed by the cosine similarity measure (Salton 97).

**(e) Similarity to Keywords.** This feature is obtained in a way similar to the previous feature. In this case, we calculate the similarity between a vector of keywords and each sentence, using the cosine vector-similarity measure.

**(f) Sentence-to-Sentence Cohesion.** For each sentence  $s$  this feature is obtained as follows: first, one computes the similarity between  $s$  and each of the other sentences of the document; then one adds up all those values of similarity, obtaining the raw value of this feature for  $s$ ; the process is repeated for all sentences. The value of this feature for a sentence  $s$  is normalized in the range  $[0..1]$  by computing the ratio of the raw feature value for  $s$  over the largest raw feature value among all sentences in the document. Sentences with feature values closer to 1.0 have a larger degree of cohesion.

**(g) Sentence-to-Centroid Cohesion.** For each sentence this feature is obtained as follows: first, one computes the vector representing the centroid of the document, the arithmetic average over the corresponding coordinate values of all the sentences of the document; then one computes the similarity between each sentence and the centroid, obtaining the raw value of this feature for each sentence. The value is then normalized producing a normalized value in the range  $[0..1]$  for each sentence. Sentences with feature values closer to 1.0 have a larger degree of cohesion with respect to the centroid of the document, and so are supposed to better represent the basic ideas of the document.

Our trainable summarizer also use “linguistic-oriented” features. The five following features contain information about the argumentative structure of the text. First, the text is processed by an agglomerative clustering algorithm, as proposed by Yaari (Yaari 97). The basic idea of this method is that similar text segments are iteratively grouped together, in a bottom-up fashion, based on their lexical similarity, until the highest level of the clustering scheme, the entire document. Once the agglomerative-clustering tree has been produced, five features are extracted, as follows:

**(h) Depth in the tree.** For a sentence  $s$  the value of this feature is simply the depth of  $s$  in the tree.

**(i), (j), (k), (l) Four features referring to position in a given level of the tree (position 1, position 2, position 3, and position 4, for short).** In order to extract these features for a given sentence  $s$ , we identify the path from the root of the tree to the node containing  $s$ , considering only the first four depth levels of the tree. For each depth level, a feature is identified as the direction to be taken in order to follow the path to  $s$ ; since the tree is binary, the possible values for these features are: left, right and none,

the latter value means that the sentence is in a tree node having a depth lower than four.

Other employed “linguistic” features are:

**(m) Indicator of main concepts.** A binary feature, indicating whether or not a sentence captures the main concepts of the document. The identification of these main “concepts” assumes that the majority of relevant words is a noun. Hence, for each sentence, all its nouns are identified, by using a part-of-speech software (Brill 92). This procedure is repeated for all sentences. Then, for each noun, the system computes the number of sentences in which that noun occurs. The 15 nouns with largest occurrence are selected as “main concepts”. Finally, for each sentence the value of this feature will be true if the sentence contains at least one of those 15 nouns, and false otherwise.

**(n) Occurrence of proper names.** This is also a binary feature, taking on the value true if the sentence contains at least one proper name, and false otherwise. The motivation for this feature is that the occurrence of proper names of people and places are clues that a sentence is relevant for the summary, particularly for some kinds of text. The detection of proper names was performed by a part-of-speech software (Brill 92).

**(o) Occurrence of anaphors.** In general, anaphors indicate the presence of non-essential information in a text. The detection of anaphors was performed in a way similar to the one proposed by Strzalkowski (Strzalkowski 99). The system determines whether or not certain words, characterizing anaphors, occur in the first six words of a sentence. This is also a binary feature, taking on the value true if the sentence contains at least one anaphor, and false otherwise.

**(p) Occurrence of other words indicating non-essential information.** Some other words are considered indicators of non-essential information. The words in question are speech markers such as “because”, “furthermore”, and “additionally”, which typically occur in the beginning of a sentence. This is also a binary feature, taking on the value true if the sentence contains at least one of these speech markers, and false otherwise.

## 4.2 The framework for the trainable summarizer

The trainable-summarization framework consists of the following steps. First, we apply

some standard preprocessing information retrieval methods to each document. The methods in question are removal of stop words, case folding and stemming. We have used the stemming algorithm proposed by Porter (Porter 97). Next the sentences in the document are converted to a vectorial representation (Salton 97).

After this basic preprocessing, the features described in the previous subsection are computed. Continuous features are discretized. We did some preliminary experiments with two kinds of discretization method, a simple class-blind method and a class-driven method. Surprisingly, in our preliminary experiments we did not notice any significant difference in the performance of the two discretization methods. Hence, we decided to use the simple equal-width method in our experiments.

Once all the features were duly computed and discretized, the classification algorithm was called and employed as usual in the data mining literature, namely trained on a training set and evaluated on a separate test set.

Of course, the framework assumes that each document has an extractive summary, whose sentences have the role of “positive class” instances in classification / data mining terminology. In our experiments each document’s extractive summary was automatically obtained by using an author-provided non-extractive summary as in Mani (Mani 98b).

## 5 Computational results

All the experiments were performed with documents available in the TIPSTER document base (Harman 94). The document collection consist of texts published in magazines about computers, hardware, software, etc., which have sizes varying from 2 Kbytes to 64 Kbytes. We have used in our experiments only documents which already had a summary provided by the own author of the document, in order to use the above-mentioned technique of automatically producing an extractive summary. The TIPSTER document base contained 33,658 documents with the author’s manual summary. A subset of these documents was randomly selected for the experiments to be reported in this section.

In our experiments with the trainable summarizer, we have used two very well-known classification algorithms, namely Naive Bayes

(Mitchell 97) and C4.5 (Quinlan 92). The former is a Bayesian classifier that assumes that the features are independent from each other. Despite this unrealistic assumption, the method presents good results in practice in many cases, and it has been used in many text mining projects. C4.5 is a decision-tree algorithm that is often used for comparison purposes with other classification algorithms, particularly in the data mining and machine learning communities.

In total the experiments involved five text-summarization methods, as follows:

(a) Our trainable summarizer using C4.5 as the classifier.

(b) Our trainable summarizer using Naive Bayes as the classifier.

(c) Random Summarizer. This method randomly selects  $n$  sentences from the text, where  $n$  is determined by the desired compression rate. This system provides a very simple, weak baseline for the performance of any text-summarization method.

(d) First Sentences. This method selects the first  $n$  sentences of the document, where  $n$  is determined by the desired compression rate. This system provides a simple, yet relatively strong, baseline for the performance of any text-summarization method Brandow (Brandow 94).

(e) Word Summarizer. Microsoft’s Word Summarizer is a text summarizer which is part of Microsoft Word, and it has been used for comparison with other summarization methods by several authors (Barzilay 97), (Marcu 99). This method uses non-documented techniques to perform an “almost extractive” summary from a text, with a compression rate determined by the user.

This method has a couple of characteristics that are different from all the previous methods, as follows. First, the compression rate specified by the user refers to the number of characters to be extracted, and not to the number of sentences. Second, some sentences are somewhat modified by Word Summarizer during its execution, in order to reduce the sentence’s number of characters without sacrificing its comprehensibility. These characteristics constituted a problem for our experiments, since due to them a comparison between Word Summarizer and the other methods is not entirely fair: (i) due to the first characteristic, the summaries produced by Word Summarizer can contain a few more or a few less sentences than the summaries produced by the other text-summarization methods; (ii) due to the second

characteristic, in some cases it will not be possible to compute an exact match between a sentence selected by Word Summarizer and an original sentence of the document; these cases are rare, and the corresponding sentences were ignored.

Note that only our proposal is a trainable summarizer, the remaining three methods are not trainable, and were used mainly as baselines for comparison with the trainable methods.

The set of documents used in our first experiment consisted of 200 documents, partitioned into disjoint training and test sets with 100 documents each. The training set contained 25 documents of 11 Kbytes, 25 documents of 12 Kbytes, 25 documents of 16 Kbytes, and 25 documents of 31 Kbytes. The average number of sentences per document is 129.5, since there are in total 12,950 sentences in the training set. The test set contained 25 documents of 10 Kbytes, 25 documents of 13 Kbytes, 25 documents of 15 Kbytes, and 25 documents of 28 Kbytes. The average number of sentences per document is 118.6, since there are in total 11,860 sentences in the test set.

Table 1 reports the results obtained by the five summarizers. The values of precision/recall are expressed in terms of percentage (%). The figures after the “ $\pm$ ” symbol are standard deviations. The best results are shown in boldface.

Let us now analyze the results reported in Table 1. For all the five methods, the values of precision and recall are significantly higher with the compression rate of 20% than with the compression rate of 10%. This result was expected, since larger the compression rate, the larger the number of sentences to be selected for the summary, and so the larger the probability that a sentence selected by a summarizer matches with a sentence belonging to the extractive summary.

For the compression rates of 10% and 20 % the best results were obtained by our trainable summarizer with Naive Bayes classifier.

We can also note that using the same features, but with the C4.5 as classifier, the obtained results were poor in comparison with the First-Sentences and Word Summarizer baselines.

Perhaps this result offers a useful lesson: most projects on trainable summarizers focus on the proposal of new features for classification, using more and more elaborate statistics-based or linguistics-based features, but they usually

use a single classifier in the experiments. Normally “conventional” classifiers are used. Our results suggest that researches should pay more attention to the development of more elaborate classifiers, tailored for text-summarization applications, or at least choose the best classifier among the conventional ones already available.

**Table 1: Results for documents with an automatically-produced extractive summary**

	Compression rate: 10%		Compression rate: 20%	
	Precision	Recall	Precision	Recall
Trainable-S-C4.5	22.4 $\pm$ 1.48	22.4 $\pm$ 1.48	34.7 $\pm$ 1.01	34.6 $\pm$ 1.03
Trainable-S-Bayes	<b>40.5 <math>\pm</math> 1.99</b>	<b>40.5 <math>\pm</math> 1.99</b>	<b>51.4 <math>\pm</math> 1.47</b>	<b>51.4 <math>\pm</math> 1.47</b>
Random-Summ.	8.9 $\pm$ 0.94	8.9 $\pm$ 0.94	19.6 $\pm$ 0.88	19.6 $\pm$ 0.88
First-Sentences	23.9 $\pm$ 1.60	23.9 $\pm$ 1.60	32.0 $\pm$ 1.36	32.0 $\pm$ 1.36
Word-Summ.	26.1 $\pm$ 1.21	34.4 $\pm$ 1.56	38.8 $\pm$ 1.14	43.7 $\pm$ 1.30

## 6 Conclusions and future research

In this paper we have explored the framework of trainable text summarizers, which was proposed a few years ago by Kupiec (Kupiec 95). Our choice of this framework was motivated by the fact that it allows us to measure the results of a text summarization algorithm in an objective way, similarly to the standard evaluation of classification algorithms. This avoids the difficult problem of subjective evaluation of the quality of a summary.

The main contribution of this paper is that it performs a more extensive investigation of that framework. We proposed a trainable summarizer which employs statistics-oriented and linguistics-oriented features, used with two different classification algorithms, Naive Bayes and C4.5. Hence, we were able to analyze the performance of two different text-summarization methods. The performance of these methods was also compared with the performance of three non-trainable, baseline methods.

In general the trainable method using Naive Bayes classifier significantly outperformed all the three baseline methods.

The most interesting finding of our experiments was that the choice of the classifier (Naive Bayes versus C4.5) had severely influenced the performance of the trainable summarizer. In our future research we intend to focus mainly on the development of a new or extended classification algorithm tailored for text summarization. In addition, experiments are in hand to evaluate the performance of our trainable summarizer in documents where an extractive summary is directly provided by the author, rather than being automatically generated as in the experiments reported in this paper.

## References

- (Barzilay 97) Barzilay, R. ; Elahad, M. Using Lexical Chains for Text Summarization. In Mani, I. e Maybury, M. T. (eds.). In *Proceedings of the ACL/EACL-97 Workshop on Intelligent Scalable Text Summarization*, Association of Computational Linguistics, 1997.
- (Brandow 94) Brandow, R.; Mitze, K., Rau, L. Automatic condensation of electronic publications by sentence selection. *Information Processing and Management* 31(5) , 675-685, 1994.
- (Brill 92) Brill, E. A simple rule-based part-of-speech tagger. In *Proceedings of the Third Conference on Applied Computational Linguistics*. Association for Computational Linguistics, 1992.
- (Carbonell 98) Carbonell, J. G.; Goldstein, J. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of SIGIR-98*, 1998.
- (Edmundson 69) Edmundson, H. P. New methods in automatic extracting. *Journal of the Association for Computing Machinery* 16 (2),264-285, 1969.
- (Harman 94) Harman, D. Data Preparation. In Merchant, R. (ed.). *The Proceedings of the TIPSTER Text Program Phase I*. Morgan Kaufmann Publishing Co., 1994.
- (Kupiec 95) Kupiec, J. ; Pedersen, J. O.; Chen, F. A trainable document summarizer. In *Proceedings of the 18th ACM-SIGIR Conference, Association of Computing Machinery, Special Interest Group Information Retrieval*, 68-73. 1995.
- (Larocca 00) Larocca Neto, J.; Santos, A. D.; Kaestner, C.A.; Freitas, A.A.. Document clustering and text summarization. *Proceedings of 4th Int. Conf. Practical Applications of Knowledge Discovery and Data Mining (PADD-2000)*, 41-55, London: The Practical Application Company, 2000.
- (Luhn 58) Luhn, H. The automatic creation of literature abstracts. *IBM Journal of Research and Development* 2(92), 159-165, 1958.
- (Mani 98a) Mani, I.; House, D.; Klein, G.; Hirschman, L.; Obrsl, L.; Firmin, T.; Chrzanowski, M.; Sundheim, B. *The TIPSTER SUMMAC Text Summarization Evaluation*. MITRE Technical Report MTR 98W0000138. The MITRE Corporation, Oct. 1998.
- (Mani 98b) Mani, I.; Bloedorn, E. Machine Learning of Generic and User-Focused Summarization. In *Proceedings of the Fifteenth National Conference on AI (AAAI-98)*, 821-826, 1998.
- (Marcu 99) Marcu, D. Discourse trees are good indicators of importance in text. In I. Mani., I. ; Maybury, M. (eds.). *Advances in Automatic Text Summarization*, 123-136. The MIT Press, 1999.
- (Mitchell 97) Mitchell, T. *Machine Learning*. McGraw-Hill, 1997.
- (Mitra 97) Mitra, M.; Singhal, A.; Buckley, C. Automatic text summarization by paragraph extraction. In *Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization*. Madrid, Spain, 1997.
- (Nevill-Manning 99) Nevill-Manning, C. G. ; Witten, I. H. Paynter, G. W. et al. KEA: Practical Automatic Keyphrase Extraction. *ACM DL 1999*: 254-255, 1999.
- (Porter 97) Porter, M.F. An algorithm for suffix stripping. *Program* 14, 130-137. 1980. Reprinted in: Spark-Jones, K.; Willet, P. (eds.) *Readings in Information Retrieval*, 313-316. Morgan Kaufmann, 1997.
- (Quinlan 92) Quinlan, J. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, Sao Mateo, CA, 1992.
- (Rath 61) Rath, G. J. ; Resnick A. ; Savvage R. The formation of abstracts by the selection of sentences: Part 1: sentence selection by man and machines. *American Documentation* 12 (2), 139-141, 1961.

(Salton 97) Salton, G.; Buckley, C. Term-weighting approaches in automatic text retrieval. *Information Processing and Management* 24, 513-523. 1988. Reprinted in: Sparc-Jones, K.; Willet, P. (eds.) *Readings in Information Retrieval*, 323-328. Morgan Kaufmann, 1997.

(Spark-Jones 99) Spark-Jones, K. Automatic summarizing: factors and directions. In Mani, I.; Maybury, M. *Advances in Automatic Text Summarization*, 1-12. The MIT Press, 1999.

(Strzalkowski 99) Strzalkowski, T.; Stein, G.; Wang, J.; Wise, B. A Robust Practical Text Summarizer. In Mani, I.; Maybury, M. (eds.), *Advances in Automatic Text Summarization*. The MIT Press, 1999.

(Teufel 99) Teufel, S.; Moens, M. Argumentative classification of extracted sentences as a first step towards flexible abstracting. In Mani, I.; Maybury, M. (eds.), *Advances in automatic text summarization*, The MIT Press, 1999.

(Turney 00) Turney, P.D. Learning algorithms for keyphrase extraction. *Information Retrieval*, 2 (4), 2000.

(Weiss 98) Weiss, S.W.; Indurkha, N. *Predictive Data Mining*. Morgan Kaufmann, 1998.

(Yaari 97) Yaari, Y. Segmentation of Expository Texts by Hierarchical Agglomerative Clustering. *Technical Report*, Bar-Ilan University, Israel, 1997.